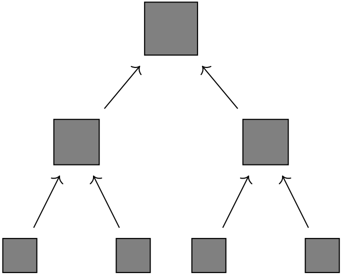
Reducing communications and memory costs of a parallel Block Low-Rank solver

P. Amestoy[1] O. Boiteau[2] A. Buttari[3] M. Gerest[2,4]
F. Jézéquel[4] J.-Y. L'Excellent[1] T. Mary[4]
[1]Mumps Technologies [2]EDF R&D [3]CNRS-IRIT [4]Sorbonne Université-CNRS-LIP6

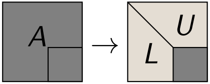
MUMPS User Days, 22 June 2023

The multifrontal method

- Solving sparse linear system $Ax = b$
  - Factorization $A = LU$
  - Solve triangular systems $Ly = b$ and $Ux = y$
- Multifrontal method: we need to compute partial LU factorizations of several dense matrices



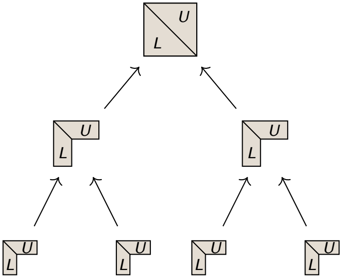
Matrix A (elimination tree)



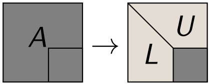
Frontal matrix at each node: partial LU factorization

The multifrontal method

- Solving sparse linear system $Ax = b$
  - Factorization $A = LU$
  - Solve triangular systems $Ly = b$ and $Ux = y$
- Multifrontal method: we need to compute partial LU factorizations of several dense matrices



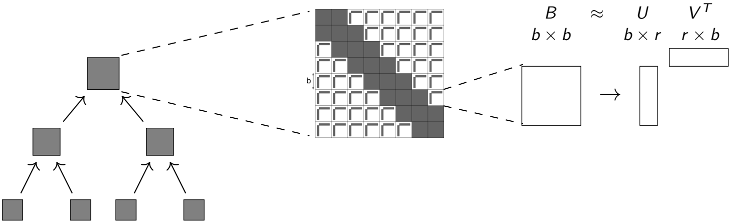
L and U factors



Frontal matrix at each node: partial LU factorization

BLR compression

- Block Low-Rank (BLR) compression of a dense matrix: we try to compress the off-diagonal blocks:

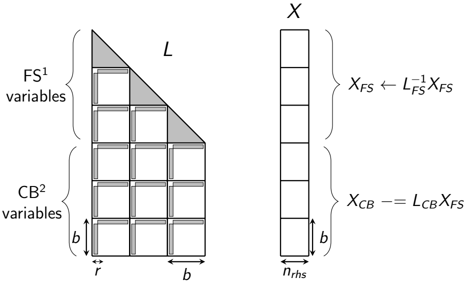


- Low-rank approximation with accuracy ε, controlled by the user
- U, V, and the rank r are chosen so that $\|B - UV^T\| \leq \varepsilon$
- Example: truncated SVD or truncated QR decomposition

P. Amestoy, C. Ashcraft, O. Boiteau, A. Buttari, J.-Y. L'Excellent, and C. Weisbecker. *"Improving Multifrontal Methods by Means of Block Low-Rank Representations"*. SIAM SISC (2015).

Triangular solve

- **Forward elimination**: solve $Lx = b$, bottom-up traversal of the elimination tree
- We solve a triangular system at each node
- Possibly several right-hand sides $\rightarrow$ X has n_{rhs} columns



A BLR frontal matrix and its right-hand sides X

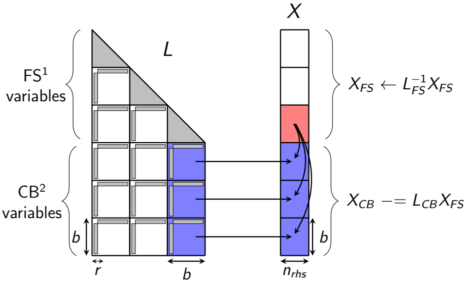
Right-looking algorithm

for $j \in FS$ **do**
  $X_j \leftarrow L_{jj}^{-1} X_j$
  for $i > j$ **do**
    $X_i \leftarrow X_i - U_{ij}(V_{ij}^T X_j)$
  end for
end for

[1] FS: Fully-Summed variables

[2] CB: Contribution Block, to be eliminated in another front

- **Forward elimination**: solve $Lx = b$, bottom-up traversal of the elimination tree
- We solve a triangular system at each node
- Possibly several right-hand sides $\rightarrow$ X has n_{rhs} columns



"Right-looking" algorithm, step $j = 3$

Right-looking algorithm

for $j \in FS$ **do**
  $X_j \leftarrow L_{jj}^{-1} X_j$
  for $i > j$ **do**
    $X_i \leftarrow X_i - U_{ij}(V_{ij}^T X_j)$
  end for
end for

[1]FS: Fully-Summed variables
[2]CB: Contribution Block, to be eliminated in another front

Objectives

- Motivation: the BLR triangular solve is memory-bound
- Objective: Reduce data movements → obtain time reduction

[1] https://mumps-solver.org

[2] https://www.calmip.univ-toulouse.fr/

Objectives

- Motivation: the BLR triangular solve is memory-bound
- Objective: Reduce data movements $\rightarrow$ obtain time reduction

- Outline:
  - Reduce data access **to the factors**, using mixed precision
  - Reduce data access **to the right-hand sides** (RHS), changing the order of the operations

[1] https://mumps-solver.org
[2] https://www.calmip.univ-toulouse.fr/

Objectives

- Motivation: the BLR triangular solve is memory-bound
- Objective: Reduce data movements $\rightarrow$ obtain time reduction

- Outline:
  - Reduce data access **to the factors**, using mixed precision
  - Reduce data access **to the right-hand sides** (RHS), changing the order of the operations

- Implementation in multifrontal solver MUMPS[1]
- Validation on industrial problems
- Experiments done on Olympe supercomputer (CALMIP[2], Toulouse)

---

[1] https://mumps-solver.org

[2] https://www.calmip.univ-toulouse.fr/

BLR in mixed precision

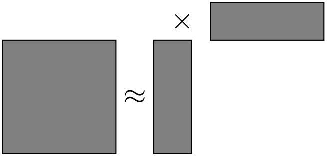
A standard (uniform precision) low-rank approximation $B \approx UV^T$

- In a low-rank approximation, the last columns may be stored in lower precision ($\rightarrow$ small singular values)
- Same level of accuracy
- See article:

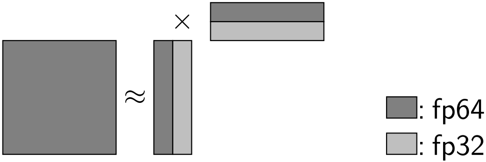
P. Amestoy, O. Boiteau, A. Buttari, M. Gerest, F. Jézéquel, J.-Y. L'Excellent, and T. Mary. *"Mixed Precision Low-Rank Approximations and Their Application to Block Low-Rank Matrix Factorization"*. IMAJNA (2022).

$\rightarrow$ Reduced factor size in memory

BLR in mixed precision



A standard (uniform precision) low-rank approximation
$B \approx UV^T$



A low-rank approximation in mixed precision

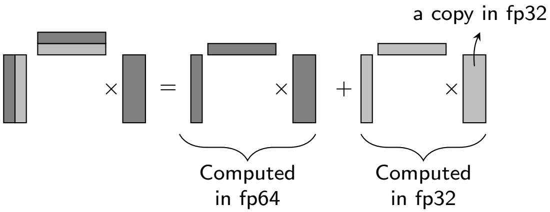
- In a low-rank approximation, the last columns may be stored in lower precision ($\rightarrow$ small singular values)
- Same level of accuracy
- See article:

P. Amestoy, O. Boiteau, A. Buttari, M. Gerest, F. Jézéquel, J.-Y. L'Excellent, and T. Mary. *"Mixed Precision Low-Rank Approximations and Their Application to Block Low-Rank Matrix Factorization"*. IMAJNA (2022).

$\rightarrow$ Reduced factor size in memory

Product LR block × RHS in mixed precision

- The main kernel in BLR solve: product LR block × RHS



- Most operations are now switched to lower precision

- Mixed precision requires extra accesses to the RHS (a copy in each precision + extra conversion operations)

Mixed precision: storage gains

			factor size (% of FR)		
Matrix	N	Factor entries (full-rank)	BLR double	BLR mixed	gain mixed vs double
lfm_aug5M	6E+6	13E+9	46.4%	33.2%	-28%
Queen_4147	4E+6	14E+9	51.8%	39.0%	-25%
Thmgaz	5E+6	18E+9	67.4%	49.8%	-26%
Poisson200	8E+6	30E+9	22.8%	15.8%	-31%
Electrophys	10E+6	22E+9	19.5%	15.7%	-19%

Table: storage gains from mixed precision in MUMPS

- BLR in mixed precision: the memory cost of the factors is reduced
- Reduction of the factor size, up to 31%
- Matrices from SuiteSparse collection and MUMPS' industrial partners
- Run on 2 MPI $\times$ 18 OMP, $\varepsilon = 10^{-9}$

Mixed precision: time gains (1 RHS)

Matrix	precision for BLR	
	double	mixed
lfm_aug5M	0.23	0.18 (-22%)[3]
Queen_4147	0.36	0.30 (-16%)
Thmgaz	0.45	0.35 (-22%)
Poisson200	0.39	0.36 (-7%)
Electrophys	0.26	0.23 (-12%)

Table: Time (in seconds) spent in forward elimination

Matrix lfm_aug5M:

- Time reduction from mixed precision (double+single): 22%
- Storage reduction from mixed precision: 28%

[3] gain vs double precision BLR

[4] factorization failed because this matrix is numerically singular in single precision

Mixed precision: time gains (1 RHS)

Matrix	precision for BLR		
	double	mixed	single
lfm_aug5M	0.23	0.18 (-22%)[3]	0.13 (-41%)[5]
Queen_4147	0.36	0.30 (-16%)	0.20 (-43%)
Thmgaz	0.45	0.35 (-22%)	0.25 (-45%)
Poisson200	0.39	0.36 (-7%)	0.33 (-16%)
Electrophys	0.26	0.23 (-12%)	failed[4]

Table: Time (in seconds) spent in forward elimination

Matrix lfm_aug5M:

- Time reduction from mixed precision (double+single): 22%
- Storage reduction from mixed precision: 28%
- Time reduction from single precision: 41%

[3] gain vs double precision BLR

[4] factorization failed because this matrix is numerically singular in single precision

Mixed precision: time gains (multiple RHS)

Matrix	n_{rhs}	time for forward elimination (s)		
		BLR double	BLR mixed	gain mixed vs double
lfm_aug5M	1	0.23	0.18	-22%
	250	4.03	4.14	+3%
Queen_4147	1	0.36	0.30	-16%
	250	3.7	3.80	+2%
Thmgaz	1	0.45	0.35	-22%
	250	3.55	3.06	-14%
Poisson200	1	0.39	0.36	-7%
	30	0.56	0.54	-4%
Electrophys	1	0.26	0.23	-12%
	250	3.07	3.03	-1%

[5] Full-rank variant (FR): no BLR compression is used

[6] OOM: out of memory

Mixed precision: time gains (multiple RHS)

Matrix	n_{rhs}	time for forward elimination (s)				
		FR[5]	BLR double	gain BLR vs FR	BLR mixed	gain mixed vs double
lfm_aug5M	1	0.40	0.23	-43%	0.18	-22%
	250	5.53	4.03	-27%	4.14	+3%
Queen_4147	1	0.62	0.36	-42%	0.30	-16%
	250	5.48	3.7	-32%	3.80	+2%
Thmgaz	1	0.80	0.45	-43%	0.35	-22%
	250	OOM[6]	3.55	-	3.06	-14%
Poisson200	1	OOM	0.39	-	0.36	-7%
	30	OOM	0.56	-	0.54	-4%
Electrophys	1	OOM	0.26	-	0.23	-12%
	250	OOM	3.07	-	3.03	-1%

[5] Full-rank variant (FR): no BLR compression is used
[6] OOM: out of memory

Challenges in the case of multiple RHS

- Why are the gains not as good with multiple RHS than 1 RHS? (in both cases: BLR vs FR and mixed vs double)

- **The dominant time cost** is no longer accessing the factors, but **accessing the RHS**

Challenges in the case of multiple RHS

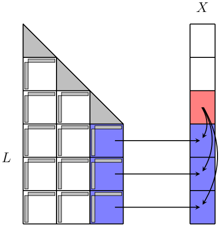
- Why are the gains not as good with multiple RHS than 1 RHS? (in both cases: BLR vs FR and mixed vs double)

- **The dominant time cost** is no longer accessing the factors, but **accessing the RHS**

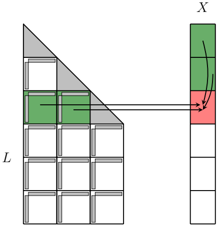
$\rightarrow$ **We need to rethink the communication patterns to minimize the data access to the RHS**

Right-looking vs Left-looking communication patterns

- Operations in triangular solve $\rightarrow$ several possible orders



Right-looking ("eager")



Left-looking ("lazy")

Left-looking (LL) algorithm

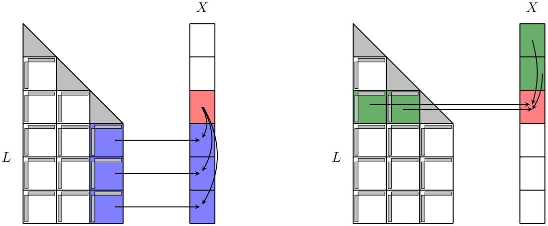
for $i \in FS$ **do** ▷ Sequential loop
  for $j < i$ **do** ▷ Parallel loop
    $X_i \leftarrow X_i - U_{ij}(V_{ij}^T X_j)$
  end for
  $X_j \leftarrow L_{jj}^{-1} X_j$
end for
for $i \in CB$ **do** ▷ Parallel loop
  for $j \in FS$ **do** ▷ Sequential loop
    $X_i \leftarrow X_i - U_{ij}(V_{ij}^T X_j)$
  end for
end for

Right-looking (RL) algorithm

for $j \in FS$ **do** ▷ Sequential loop
  $X_j \leftarrow L_{jj}^{-1} X_j$
  for $i > j$ **do** ▷ Parallel loop
    $X_i \leftarrow X_i - U_{ij}(V_{ij}^T X_j)$
  end for
end for

Right-looking vs Left-looking communication patterns

- Operations in triangular solve $\rightarrow$ several possible orders

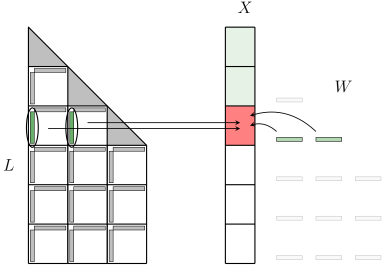


Right-looking ("eager") Left-looking ("lazy")

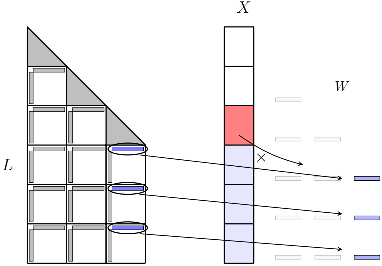
- Right-looking: at each step, one "read" operation and many "write" operations on the RHS
- Left-looking: many "reads", one "write"
- In both cases: poor data locality

Hybrid algorithm

- A hybrid algorithm, combination of right-looking and left-looking:



Step $k = 3$: Access U_{kj} for $j < k$ in left-looking and update block X_k



Step $k = 3$: Read X_k and V_{ik}^T for $i > k$ in right-looking

- The kernel $UV^T \times$ RHS is decomposed into 2 steps:
  - $W = V^T X$, done in right-looking
  - UW, done in left-looking
- Each block of the RHS is written once and used once $\rightarrow$ locality improved.
- Need to store W $\rightarrow$ OK if the ranks are small enough

Hybrid algorithm

Hybrid algorithm

for $k \in FS$ **do** ▷ Sequential loop
    for $j < k$ **do** ▷ Parallel loop (left-looking)
        $\boldsymbol{X_k} \leftarrow \boldsymbol{X_k} - U_{kj} W_{kj}$
    end for
    $\boldsymbol{X_k} \leftarrow L_{kk}^{-1} \boldsymbol{X_k}$
    for $i > k$ **do** ▷ Parallel loop (right-looking)
        $W_{ik} = V_{ik}^T \boldsymbol{X_k}$
    end for
end for
for $k \in CB$ **do** ▷ Parallel loop
    for $j \in FS$ **do** ▷ Sequential loop (left-looking)
        $\boldsymbol{X_k} \leftarrow \boldsymbol{X_k} - U_{kj} W_{kj}$
    end for
end for

- We also implemented another version of this algorithm, better suited to multicore parallelism

Communication volume analysis

- **Communication volume** between a **fast memory** (example: cache) and a **slower memory** (example: RAM)

Variant	Communication volume		
	Read Only	Write Only	Read/Write
Right-Looking (RL)	$2qrb$		qbn_{rhs}
Left-Looking (LL)	$2qrb + qbn_{\mathrm{rhs}}$		
Hybrid	$2qrb + qrn_{\mathrm{rhs}}$	qrn_{rhs}	

q: number of blocks in the factors

b: block size

- Left-looking vs Right-looking: same number of accesses (but "read only" vs "read/write")
- Relative gain hybrid vs left-looking:

$$\frac{\mathit{volume}(\text{left-looking})}{\mathit{volume}(\text{hybrid})} \approx \frac{1 + n_{\mathrm{rhs}}/(2r)}{1 + n_{\mathrm{rhs}}/b} \xrightarrow{n_{\mathrm{rhs}} \to \infty} \boldsymbol{\frac{b}{2r}}$$

$\rightarrow$ Hybrid: lower communication volume, for all BLR matrices

Hybrid algorithm: time gains

Matrix	n_{rhs}	Time (s)	
		RL	Hybrid
Queen_4147	100	1.9	1.6 (-7%)
($\varepsilon = 10^{-3}$)	250	4.5	4.4 (-3%)
	500	11.8	11.0 (-12%)
Thmgaz	100	1.7	1.5 (-13%)
($\varepsilon = 10^{-4}$)	250	5.6	4.7 (-16%)
	500	12.9	10.8 (-16%)
Poisson200	100	2.0	1.7 (-14%)
($\varepsilon = 10^{-6}$)	250	5.1	4.1 (-21%)
Helmholtz140	100	2.2	1.9 (-13%)
($\varepsilon = 10^{-3}$)	250	5.7	4.8 (-15%)
	500	12.0	10.3 (-14%)

Table: Time spent in forward elimination in MUMPS

Run on 1 MPI $\times$ 18 OMP

Conclusion

New strategies to **reduce the volume of communications** in BLR triangular solve, and improve performance:

- Improve the access **to the factors**:
  - Use BLR compression in mixed precision
  - Time reductions obtained in MUMPS, up to 22%
- Improve the access **to the right-hand sides** (if multiple RHS):
  - Reorder the operations in order to improve the data locality
  - Time reductions obtained in MUMPS, up to 21%

P. Amestoy, O. Boiteau, A. Buttari, M. Gerest, F. Jézéquel, J.-Y. L'Excellent, and T. Mary. *"Communication avoiding block low-rank parallel multifrontal triangular solve with many right-hand sides"*. submitted to SIMAX (2023).

Perspectives:

- Combine both approaches (mixed + hybrid)
- Use computations in mixed precision during the factorization (ongoing implementation)

Conclusion

New strategies to **reduce the volume of communications** in BLR triangular solve, and improve performance:

- Improve the access **to the factors**:
  - Use BLR compression in mixed precision
  - Time reductions obtained in MUMPS, up to 22%
- Improve the access **to the right-hand sides** (if multiple RHS):
  - Reorder the operations in order to improve the data locality
  - Time reductions obtained in MUMPS, up to 21%

P. Amestoy, O. Boiteau, A. Buttari, M. Gerest, F. Jézéquel, J.-Y. L'Excellent, and T. Mary. *"Communication avoiding block low-rank parallel multifrontal triangular solve with many right-hand sides"*. submitted to SIMAX (2023).

Perspectives:

- Combine both approaches (mixed + hybrid)
- Use computations in mixed precision during the factorization (ongoing implementation)

THANK YOU